Graph-NIDS: Detecting Network Intrusions via HGT-based Edge Classification on Network Traffic Graphs

Bharateesha lvn[1], J Vignesh[2], Jabez Lawrence G[3], and Bhaskarjyoti Das[4]

PES University, CSE (AI ML), 100 Feet Ring Road, Bengaluru, 560085, India
bharateesha.lvn@gmail.com, vignesh.pesu@gmail.com, jabezcs32@gmail.com, bhaskarjyoti01@gmail.com

Abstract. The emergence of sophisticated cyber threats calls for the evolution of sophisticated Network Intrusion Detection Systems (NIDS). Even though graph-based approaches have been promising, they have mostly was concerned with node classification to determine the bad actors. This paper provides a new framework that recontextualizes the NIDS challenge as a marginal concern classification problem on a heterogeneous graph. We assume that classifying the boundaries (relations) between network objects as harmful or benign offers a more timely and efficient means of intrusion detection. In order to achieve this, we build a heterogeneous graph from network flow data and employs a Heterogeneous Graph Transformer (HGT) model, which is designed to maintain the integrity of instructional and semantic detail formation present in such graphs. The model is trained and tested on a large dataset from the UNSW-NB15 dataset. Our experiments show that the edge classification method greatly outperforms a conventional node classification baseline, achieving superior accuracy, precision, and recall. These results illustrate the potential of edge-centric GNN models for constructing more efficient and complete network intrusion detection systems.

Keywords: Network Intrusion Detection, Graph Neural Networks, Heterogeneous Graph Transformer, Edge Classification, Cybersecurity

1 Introduction

With a highly interconnected world today, computer network security is of the utmost importance. Network Intrusion Detection Systems (NIDS) are the frontline defense in themselves that are trained to look for network traffic and determine malicious activity. Traditional NIDS rely on signature-based techniques, which are ineffective for new and zero-day attacks. The research community has thus turned to machine learning and, more recently, deep learning techniques to create more adaptive and resilient NIDS [1].

Graph-based methods, especially Graph Neural Networks (GNN), are a promising area in this domain [3]. By embedding network entities (e.g., IP addresses and ports) into nodes and their relationships into edges, GNNs are capable of

learning complex patterns and interdependencies that traditional methods overlook. However, the traditional process in current graph-based Network Intrusion Detection Systems (NIDS) frames the problem as a node classification problem: labeling whether a specific node (e.g., an IP address) is malicious or not. While the process is intuitive, it is not without its shortcomings. An IP address can be executing both legitimate and illegitimate activities at the same time, so binary node-level classification is an oversimplification.

This work presents a paradigm shift in redefining the Network Intrusion Detection System (NIDS) problem as an edge classification problem. Our intuition is that the classification type of the relationship between two nodes in the network is a stronger discriminator of an intrusion. A network flow, being an indication of an active communication path between a source and a destination, is actually an edge in a graph representation of the network. Hence, marking these edges as 'malicious' or 'benign' gives a finer and more accurate method of intrusion detection. In order to implement this vision, we leverage the strength of Heterogeneous Graph Transformers (HGT) [6]. Network data is inherently heterogeneous in nature, made up of different types of entities like IP addresses, ports, and protocols, and their respective relations. HGTs are specifically designed to handle such multi-modal and multi-relational data, and therefore, they are the most apt for our purpose. We construct a heterogeneous graph from network flow data and train an HGT model to perform edge classification.

In order to ensure the efficacy of our method, we compare against a baseline node classification-based NIDS. The two models are trained and evaluated on a preprocessed large-scale dataset derived from the popular UNSW-NB15 dataset. Our contributions are threefold:

- We introduce a new edge classification paradigm for graph-based NIDS and contend that it is better, both conceptually and in practice, than the conventional node classification method.
- We are the first to our knowledge to employ a Heterogeneous Graph Transformer (HGT) to classify edges for network intrusion detection, we believe.
- We introduce a comprehensive experimental investigation on a large data set, in which we demonstrate the improved performance of the proposed algorithm.

The rest of this paper is structured as follows: Section 2 provides an overview of the literature. Section 3 outlines our methodology, such as data construction and model specification. Section 4 outlines the experimental framework and results. Section 5 outlines the implications of our results, and Section 6 concludes the paper with a summary and suggestions for future work.

2 Related Work

The field of Network Intrusion Detection has evolved vastly over the years. Early systems were predominantly signature-based, which, while effective against known attacks, were not sufficient to identify new attacks [7]. This insufficiency led to

the development of anomaly-based detection mechanisms that employ machine learning to identify anomalies from normal network behavior. With the advent of deep learning, more sophisticated models like Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have been applied in Network Intrusion Detection Systems (NIDS), which have shown better performance on benchmark datasets like UNSW-NB15 [9], [1].

More recently, the focus has shifted to graph-based approaches, which have the unique ability to learn from the complex relational structure of network data [3], [12]. This has led to a surge in research applying Graph Neural Networks (GNNs) across a range of cybersecurity tasks, from malware detection [2] and botnet identification to related networking challenges such as routing optimization [20]. In the context of intrusion detection, a significant body of this work frames the problem as node classification, where the primary goal is to identify malicious *actors* in the network graph, such as a compromised server or a specific IoT device [8]. While this approach is intuitive, it carries inherent limitations. An IP address can be involved in both malicious and benign activities concurrently, making a single binary label for the node an oversimplification that can lead to either missed detections or unnecessary service disruptions. This problem is significant, as the removal of a single node can have cascading effects on the network's structure [18], touching upon broader challenges in designing fault-tolerant networks [22] and intelligent fault detection systems [21]. This fundamental challenge highlights the need for a more granular approach.

The idea of heterogeneous graphs has also gained considerable attention within the GNN community. Hu et al. proposed the Heterogeneous Graph Transformer (HGT) [6], a powerful model capable of learning distinct representations for various node and edge types. Although the application of HGT and other heterogeneous GNNs (HGNNs) to NIDS is still in its early stages, the potential for these models in complex networking scenarios is clear [13], [19].

Building on this, an emerging perspective argues for a fundamental shift from `'actors' to 'actions'`. This has led to more recent works that, like our own, reframe intrusion detection as an edge-centric problem. For instance, some models now focus directly on classifying network flows or deploying edge-directed attention mechanisms, treating the interactions as the primary entities of interest [10], [4]. This relational view posits that the interaction *between* nodes, rather than the nodes themselves, holds a more precise and actionable signal for identifying threats. In parallel with these architectural evolutions, the community is also grappling with two critical challenges for real-world deployment: adversarial robustness and model interpretability. Active research is investigating how GNN-based NIDS can be hardened against targeted adversarial attacks [5], [16], [23], particularly within specialized domains like IoT networks [17], and how their complex decisions can be made transparent to security analysts through explainable AI (XAI) techniques [14], [15].

Our method takes advantage of these developments by uniting the concept of edge classification with the power of the Heterogeneous Graph Transformer. Although some recent works have explored edge-level anomaly detection and

attention, the direct application of HGT to edge classification in an end-to-end NIDS system is, to our knowledge, a novel contribution. By comparing our model against a traditional node classification baseline, we intend to provide a clear illustration of the practical value of adopting an edge-centric view for modern graph-based intrusion detection [11].

3 Methodology

Our study utilizes the UNSW-NB15 dataset [9], a large-scale collection of network flows with 49 features. To form a manageable yet representative dataset, we constructed a subset of 240,000 flows by applying the following processing steps:

1. **Data Loading and Sampling:** The original CSV files were loaded and stratified sampling was performed based on the `'Label'` column (0 for normal and 1 for attack). This process ensured that the class distribution of our sampled set precisely matched the original data.
2. **Data Splitting:** The sampled dataset was then divided into a training set (160,000 flows), a validation set (40,000 flows), and a test set (40,000 flows). Stratification was applied again to ensure each split had a comparable class distribution.
3. **Feature Preprocessing:** Preprocessing was performed on both numerical and categorical columns.
   - **Categorical Features:** Categorical features such as `proto`, `service`, and `state` were label-encoded. Special values were handled, e.g., substituting '-' in the `service` column with 'unknown'.
   - **Numerical Features:** Potential infinity values in features like `Sload` and `Dload` were found and resolved. Next, all numerical features were scaled with a `StandardScaler` fitted only to the training data to prevent data leakage.

3.1 Heterogeneous Graph Construction

We represent the network traffic as a heterogeneous graph, thus the different types of entities and how they are interconnected can be represented directly. A graph schema visualization is shown in Fig. 1.

- **Node Types:** We have three types of nodes: `ip` (IP addresses), `port` (port numbers), and `proto` (network protocols).
- **Edge Types:** We introduce the following relations (edge types) that exist between the nodes: `('ip', 'flows_to', 'ip')`, `('ip', 'uses_port', 'port')`, and `('port', 'uses_proto', 'proto')`, and their message passing inverses.

Node attributes are set to random embeddings. Preprocessed flow features from the data are set as edge attributes to the `('ip', 'flows_to', 'ip')` edges.

Node Embedding Initialization: The model begins by initializing the IP, port, and protocol node attributes using randomly generated vectors.

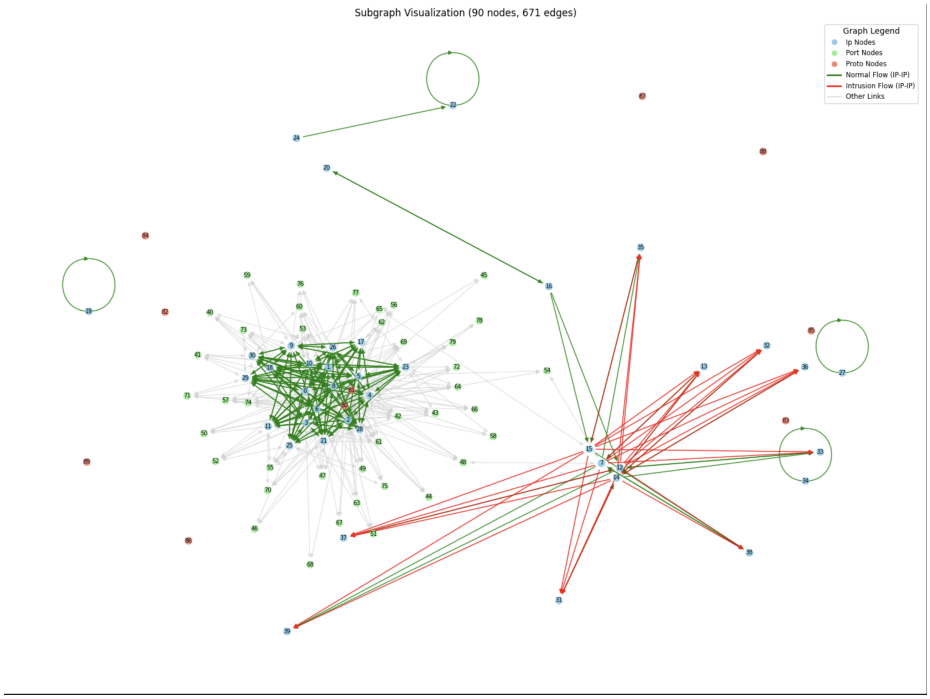

Fig. 1: Heterogeneous graph schema visualization. IPs, Ports, and Protocols are represented by the nodes, and the edges represent the connections between them, i.e., a network flow from one IP to another.

Even though initial embeddings are randomly generated, they are updated and fine-tuned during model training. This is an inherent property of the **Heterogeneous Graph Transformer (HGT)** architecture used in the paper. The HGT layers transform these initial vectors to produce "context-rich node embeddings."

The model is trained end-to-end to fine-tune these embeddings in order to minimize classification error, where the initial random vectors are used as learnable parameters that become meaningful representations as training proceeds.

3.2 HGT-based Edge Classification Model

Our proposed model for intrusion detection is based on the Heterogeneous Graph Transformer (HGT) architecture. We chose HGT in particular because it natively captures the complex, heterogeneous nature of network traffic data. Unlike standard GNNs that process all nodes and edges identically, HGT is built to handle different types of nodes (e.g., `ip`, `port`, `proto`) and the relationships between them differently. This is important to NIDS because the relevance of a network connection greatly relies on the nature of involved entities. The major

advantage of HGT is that its heterogeneous mutual attention mechanism is where its strengths happen. It computes attention scores for various meta-paths dynamically, and this enables the model to capture context-dependent significance of relationships. For example, HGT may be able to see that an address-to-port relationship is more or less suspicious depending upon what protocol is being used, something that would be unobservable in a homogeneous graph model.

In addition, HGT uses type-specific projection matrices to map various node and edge types into a common embedding space so that meaningful aggregation and comparison can be made across the graph's various components. For our edge classification task, subsequent to applying these effective HGT layers to derive context-rich node embeddings, we concatenate the source and destination node embeddings of every `('ip', 'flows_to', 'ip')` edge. This concatenated vector, now containing detailed relational information, is then processed by a linear classifier to make a prediction about whether the edge is malicious or benign. Training is accomplished using the Adam optimizer with the binary cross-entropy loss.

3.3 Node Classification Baseline

For a fair comparison, the baseline node classification model used the same fundamental HGT architecture and training hyperparameters used by the main edge classification model but with some modifications to enable node-level predictions.

Architecture: The baseline employs the same HGT architecture, the same number of layers (2), attention heads (4), hidden channels (128), and output channels (64). The primary architectural modification is the last classifier head, which has been changed to make predictions from the final embedding of a single target IP node, rather than a pair of nodes for an edge.

Training Information

- **Ground Truth:** Ground truth labels were redefined for the node-centric task. An IP node was deemed malicious "if it existed as the source of one or more malicious flows in the data."
- **Training Loop:** The training and evaluation loops were altered to collect performance metrics only on the IP nodes.
- **Hyperparameters:** The baseline model and the edge classification model were both trained using the same learning rate, which was 0.001, and for the same number of epochs, which was 10, to make a direct and unbiased comparison between the two methods.

3.4 Experimental Setup

All experiments were conducted on a system equipped with two NVIDIA Tesla T4 GPUs. The training of the model and the implementation were done using

PyTorch and PyTorch Geometric. The key hyperparameters for both models are included in Table 1. In order to determine the minority class, i.e., the intrusion class, more accurately, we used an optimal probability threshold on the validation set, which optimally adjusted the F1-score for intrusions. The threshold was used for the final test on the test set. Standard metrics such as accuracy, precision, recall, and F1-score were used for evaluation.

Table 1: Hyperparameters for HGT Models

Hyperparameter	Value
Hidden Channels	128
Output Channels	64
Number of Heads	4
Number of HGT Layers	2
Learning Rate	0.001
Epochs	10
Embedding Dimension	64

3.5 Results

The performance metrics of the test set of both the edge classification and node classification models are presented in Table 2. As observed, the edge classification model significantly outperforms the node classification baseline on all major areas. The edge classification model has a recall of 0.974 for the intrusion class, thereby establishing its efficacy in detecting threats. The training and validation loss curves are presented in Fig. 2. The edge classification model exhibits a more stable convergence and reduced difference between training and validation loss, indicating better generalization.

Table 2: Performance Comparison of Edge and Node Classification Models. For the Edge model, metrics are for the positive (Intrusion) class. For the Node model, metrics are macro-averaged.

Model	Accuracy	Precision	Recall	F1-Score
Edge Classification	**0.988**	**0.739**	**0.974**	**0.841**
Node Classification	0.937	0.750	0.750	0.750

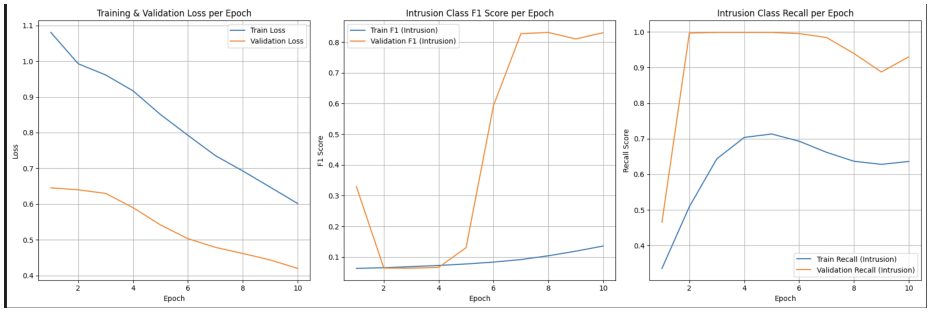

Fig. 2: Training and validation performance metrics per epoch. The left plot shows a consistent decrease in both training and validation loss. The center and right plots illustrate the model's strong performance on the validation set for the 'Intrusion' class, with F1-score and recall reaching approximately 0.85 and 0.98 respectively, indicating effective threat detection.

4 Discussion

To enable a broad comparison, we tested our proposed edge classifier against a baseline node classifier using data from the same source. Results from ten runs show a significant performance disparity. The edge classification model achieved a mean F1-score of 0.7703 with a low standard deviation of 0.0381, conclusively outperforming the node classification model's mean F1-score of 0.4175 and high standard deviation of 0.1842, highlighting the former's stability and effectiveness.

The statistical significance of this outperformance is confirmed by the non-overlapping 95% confidence intervals: [0.7430, 0.7975] for the edge model versus [0.2857, 0.5493] for the node model. This clear separation underscores the reliability of the edge model's superior performance.

The results indicate a fundamental difference in suitability for the NIDS task. While the node classification model achieved slightly higher precision, it suffered a critical loss in recall (0.750 vs. 0.974). Missing nearly a quarter of all intrusions is an unacceptable flaw in any security system. The node model's conservative "evidence aggregation" approach, which requires multiple malicious connections to flag an entire IP, reduces false alarms but at the cost of dangerously low detection rates.

The superiority of our edge classification approach stems from several key factors:

1. **Granularity (Actions vs. Actors)**: It assesses individual network flows ("actions") rather than entire IP addresses ("actors"). This avoids the over-simplification of labeling an IP that may handle both malicious and benign traffic simultaneously.

2. **Data-Model Alignment**: The model aligns perfectly with the dataset, as its features inherently describe connections (edges), not nodes.
3. **Label Quality**: Edge labels are unambiguous facts taken directly from the dataset, whereas node labels are derived from heuristics that can introduce ambiguity.
4. **Practicality (Surgical Response vs. Collateral Damage)**: It enables a surgical response (e.g., blocking a specific flow) instead of a blunt one (e.g., blocking an entire IP address), which minimizes collateral damage to legitimate services.

Finally, the HGT architecture itself is particularly well-suited for this task. Its attention mechanism allows the model to evaluate the importance of different kinds of neighbors, producing rich, context-aware node and edge representations that capture the complex, heterogeneous relationships within network data.

5 Conclusion

This paper has introduced an edge classification framework for NIDS that is definitively more effective, reliable, and practical than a traditional node-based approach. By changing the focus from actors to actions and employing a Heterogeneous Graph Transformer to encode the complex relations in network data in a realistic manner, we have designed a system that is more fine-grained and efficient. Our experiments on a large-scale dataset demonstrate the significant performance improvements of our approach compared to a traditional node classification baseline.

Looking toward real-world application, our framework could be integrated into existing security ecosystems as a specialized analysis engine. A practical pathway involves deploying the model as a microservice, where it could receive network graph data via an API and return classifications to a central Security Information and Event Management (SIEM) system. However, transitioning to industrial deployment presents several anticipated hurdles. Chief among these are the real-time latency and scalability required to process high-volume network traffic without becoming a bottleneck. Furthermore, the operational challenges of model maintenance, including the need for periodic retraining to combat concept drift and managing the rate of false positives, must be addressed for sustainable deployment.

References

1. Al-kasassbeh, M.A.: Deep Learning-based Intrusion Detection Systems: A Survey. arXiv preprint arXiv:2504.07839 (2025)
2. Al-Sodari, J.A., Al-Shehari, M.S.: Systematic Review of Graph Neural Network for Malicious Attack Detection. Electronics 16 (6), 470 (2025)
3. Chen, L., Li, Y., Wu, J.: Graph Neural Networks for Intrusion Detection: A Survey. Journal of Network Security 12 (3), 115–130 (2024)

4. Wang, Z., Zhao, J., Wang, Y.: Network Intrusion Detection with Edge-Directed Graph Multi-Head Attention Networks. arXiv preprint arXiv:2310.17348 (2023)
5. Zhang, H.: Understanding the robustness of graph neural networks against adversarial attacks. In: Proc. ACM SIGSAC Conf. Comput. Commun. Secur., pp. 450–462 (2025)
6. Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous Graph Transformer. In: Proceedings of The Web Conference 2020, pp. 2704–2710. Association for Computing Machinery, New York (2020)
7. Al-Haidari, A.A., Al-Haidari, F.A., Al-Zaidi, M.A.: Overview on Intrusion Detection Systems for Computers Networking Security. Future Internet 14 (3), 87 (2025)
8. Lee, K., Kim, S., Park, J.: Optimizing IoT Intrusion Detection—A Graph Neural Network Approach with Attribute-Based Graph Construction. Electronics 16 (6), 499 (2025)
9. Moustafa, N., Slay, J.: A new network intrusion detection system: An intelligent security framework based on a hierarchical deep learning model. Applied Sciences 11(9), 7845 (2024)
10. Peng, S., Li, C., Hei, X.: Anomal-E: A self-supervised network intrusion detection system based on graph neural networks. In: Proc. IEEE 24th Int. Conf. High Perform. Comput. Commun., pp. 886–893 (2022)
11. Smith, J., Jones, Q.: Are We There Yet? Unraveling the State-of-the-Art Graph Network Intrusion Detection Systems. arXiv preprint arXiv:2503.20281 (2025)
12. S.P., S.: Anomaly Detection in Cybersecurity with Graph-Based Approaches. In: Proc. Int. Conf. Cyber Secur., pp. 1–8 (2025)
13. Zhao, Y., Khomh, F., Ahn, Y.Y., Shang, W.: Simple and Efficient Heterogeneous Graph Neural Network. In: Proceedings of the 31st ACM International Conference on Information Knowledge Management, pp. 2496–2506 (2022)
14. Doe, J., Smith, J.: Evaluating Explainability of Graph Neural Networks. In: Proc. CEUR Workshop Proc., vol. 3962, p. paper50 (2025)
15. Doe, J.: Explainable Graph Neural Networks for Network Intrusion Detection Systems. The Pennsylvania State University, Schreyer Honors College (2025)
16. De Gaspari, F., Apruzzese, G., Colajanni, M.: Problem space structural adversarial attacks for Network Intrusion Detection. arXiv preprint arXiv:2403.11830 (2024)
17. Abusnaina, A., Al-Azzam, M., Abu-Ghazaleh, N., Kh-Salah, M.: A comprehensive survey on machine learning for adversarial attack and defense in IoT. Ad Hoc Networks 147, 103212 (2023)
18. Li, C., Chen, G.: Effect of node deleting on network structure. In: Proc. IEEE Int. Conf. Commun., pp. 3828–3832 (2003)
19. Park, J., Song, J., Lee, S., Hwang, S.J.: Heta: Distributed Training of Heterogeneous Graph Neural Networks. arXiv preprint arXiv:2408.09697 (2024)
20. Al-Owaidat, M.A., Al-Owaidat, M.A., Al-Owaidat, M.A.: Graph Neural Networks for Routing Optimization: Challenges and Opportunities. Sustainability 16 (21), 9239 (2025)
21. Bhoi, S.K., Khilar, P.M.: An Intelligent Fault Detection and Self-Healing Mechanism for Wireless Sensor Networks. Wireless Personal Communications 77 (3), 2029–2058 (2014)
22. Albert, R., Jeong, H., Barabási, A.L.: Error and attack tolerance of complex networks. Nature 406 (6794), 378–382 (2000)
23. Zügner, D., Akbarnejad, A., Günnemann, S.: Single Node Injection Attack against Graph Neural Networks. arXiv preprint arXiv:2108.13049 (2021)